Error Analysis of a Partial Pivoting Method for Structured Matrices[*]

Douglas R. Sweet
Maritime Operations Division
Defence Science and Technology Organisation
Salisbury, SA 5108
Doug.Sweet@dsto.defence.gov.au

Richard P. Brent
Computer Sciences Laboratory
Australian National University
Canberra, ACT 0200
rpb@cslab.anu.edu.au

Report TR-CS-95-03
7 June 1995

Abstract

Many matrices that arise in the solution of signal processing problems have a special *displacement structure*. For example, adaptive filtering and direction-of-arrival estimation yield matrices of Toeplitz type. A recent method of Gohberg, Kailath and Olshevsky (GKO) allows fast Gaussian elimination with partial pivoting for such structured matrices. In this paper, a rounding error analysis is performed on the Cauchy and Toeplitz variants of the GKO method. It is shown the error growth depends on the growth in certain auxiliary vectors, the *generators*, which are computed by the GKO algorithms. It is also shown that in certain circumstances, the growth in the generators can be large, and so the error growth is much larger than would be encountered with normal Gaussian elimination with partial pivoting. A modification of the algorithm to perform a type of row-column pivoting is proposed which may ameliorate this problem.

Keywords: Structured matrices, fast algorithms, displacement rank, generators, pivoting, error analysis, stability

1 Introduction

Many problems which occur in signal processing, control theory and interpolation lead to a square or rectangular system with a special structure, for which the exact or least-squares solution is required. For example, adaptive filtering requires either the exact solution of a square Toeplitz system or the least-squares solution of a rectangular Toeplitz system. A *Toeplitz* matrix is one whose entries along the NW to SE diagonals are constant, i.e. element t_{ij} depends only on $i - j$. Other types of structured matrices which arise are *Hankel* matrices whose entries along the SW to NE diagonals are constant, *Vandermonde* matrices whose entries have the form $v_{ij} = x_i^{j-1}$, and *Cauchy* matrices whose entries have the form $c_{ij} = 1/(t_i - s_j)$, where the t_i and s_j are the elements of vectors $\mathbf{t}$ and $\mathbf{s}$.

Normally, the exact or least-squares solution of a linear system requires $O(n^3)$ operations to solve, where n is the order of the system. However, the structure of the systems mentioned above has been exploited in the past [1, 4, 7] to derive *fast* solvers, i.e. those that require $O(n^2)$ or fewer operations. These fast algorithms are in general numerically unstable for indefinite systems [3, 5, 11]. Recently, methods have been proposed [6, 10, 11] which are numerically

[*] Shorter version to appear in *Advanced Signal Processing Algorithms*, Proc. SPIE 40th Annual Meeting, San Diego, July 1995.

stable, but which attempt to retain the $O(n^2)$ complexity. However, all of these algorithms will require $O(n^3)$ operations in the worst case. The BBH method [2] requires $O(n^2)$ operations in the worst case and can be shown to be *weakly stable*, but not stable in the usual sense of backward error analysis. Thus there is an interest in fast algorithms which require $O(n^2)$ operations in the worst case and can be shown to be stable.

Recently, Gohberg, Kailath and Olshevsky [8] have shown how to perform Gaussian elimination in a fast way with matrices with a special *displacement structure*. Such matrices include Toeplitz, Vandermonde, Hankel and Cauchy matrices, and generalizations thereof, called *Toeplitz type*, etc. They also show how to incorporate partial pivoting into the Cauchy and Vandermonde solvers. They point out that although pivoting cannot be incorporated directly into the corresponding Toeplitz or Hankel solvers, the Toeplitz and Hankel problems can be transformed by simple orthogonal operations into Cauchy problems. The solution to the original systems can be recovered from those of the transformed systems by the reverse orthogonal operations. Thus fast Gaussian elimination with partial pivoting can be carried out on Toeplitz, Vandermonde, Hankel and Cauchy systems.

It might be assumed that such fast solvers should have the same stability properties as Gaussian elimination with partial pivoting. One of the aims of this paper is analyse the error behaviour of these algorithms by means of a backward error analysis. It is shown that error propagation depends on the magnitude of both the triangular factors L and U (as in Gaussian elimination) and the *generators*, auxiliary vectors which are computed during the course of the algorithm.

It is shown that in some cases the generators can suffer a large growth and cause a corresponding growth in the backward and forward error. A modification is proposed which may prevent this growth, and so restore the stability of the algorithm in these cases. However, we can not prove that the modification is always successful.

The paper is structured as follows. In §2, the Gohberg-Kailath-Olshevsky (GKO) algorithm for Cauchy and Toeplitz matrices is briefly described. The error analyses of the Cauchy and Toeplitz variants of the GKO algorithm are carried out in §3 and §4 respectively. In §5, examples for both variants are given where a large growth occurs in the generators and hence in the errors in the solutions. The modified version of the GKO algorithm is proposed in §6, and numerical tests of this are carried out there. Some conclusions are drawn and suggestions for future work are given in §7.

Notation. The following notation is used. ϵ is the machine epsilon, and n is the order of the matrix to be factorized. Scalars of the form c_i and k_i are small constants. $\mathbf{e}_j$ denotes the jth column of the identity matrix. Elementwise matrix multiplication is denoted by the centred circle $\circ$. For a matrix A, $|A|$ is the matrix of moduli of the $\{a_{ij}\}$, A^I denotes elementwise inversion, and A' denotes augmentation of A to order n by adding zero rows and zero columns respectively above and to the left of A. Other submatrices are indicated in MATLAB style, i.e. for a matrix A, $A_{p:q,r:s}$ selects rows p to q of columns r to s, and a colon without an index range selects all of the rows or columns.

2 The Gohberg-Kailath-Olshevsky (GKO) Algorithm

In this section, we first define the displacement operator, displacement equation and displacement rank for structured matrices; we then give the general Gaussian elimination algorithm for structured matrices, followed by the variants for Cauchy and Toeplitz matrices.

2.1 Displacement structure

Gohberg *et al* [8] show that structured matrices satisfy a *Sylvester equation* which has the form

$$\nabla_{\{A_f,A_b\}}(R) = A_f R - R A_b = \Phi\Psi \ , \tag{1}$$

where A_f and A_b have some simple structure (usually banded, with 3 or fewer full diagonals), Φ and Ψ are $n \times \alpha$ and $\alpha \times n$ respectively, and α is some small integer (usually 4 or less). The pair of matrices Φ, Ψ is called the $\{A_f, A_b\}$-*generator* of R, and α is called the $\{A_f, A_b\}$-*displacement rank* of R.

Particular choices of A_f and A_b lead to definitions of basic classes of matrices. Thus, for a Cauchy matrix

$$C(\mathbf{t},\mathbf{s}) = \left[\frac{1}{t_i - s_j}\right]_{ij} \ ,$$

we have

$$A_f = D_t = \operatorname{diag}(t_1, t_2, \ldots, t_n), \quad A_b = D_s = \operatorname{diag}(s_1, s_2, \ldots, s_n) \tag{2}$$

and

$$\Phi^T = \Psi = [1, 1, \ldots, 1] \ .$$

More general matrices, where A_f and A_b are as in (2) but Φ and Ψ are general rank-α matrices, are called *Cauchy-type*.

Similarly, for a Toeplitz matrix $T = [t_{ij}] = [a_{i-j}]$

$$A_f = Z_1 = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & & & 0 \\ 0 & 1 & & & \vdots \\ \vdots & & \ddots & & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}, \quad A_b = Z_{-1} = \begin{bmatrix} 0 & 0 & \cdots & 0 & -1 \\ 1 & 0 & & & 0 \\ 0 & 1 & & & \vdots \\ \vdots & & \ddots & & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}, \tag{3}$$

$$\Phi = \begin{bmatrix} 1 & 0 & \cdots & \cdots & 0 \\ a_0 & a_{1-n} + a_1 & \cdots & a_{-2} + a_{n-2} & a_{-1} + a_{n-1} \end{bmatrix}^T \tag{4}$$

and

$$\Psi = \begin{bmatrix} a_{n-1} - a_{-1} & a_{n-2} - a_{-2} & \cdots & a_1 - a_{1-n} & a_0 \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix} . \tag{5}$$

2.2 Gaussian elimination for structured matrices

Let the input matrix, R_1, have the partitioning $R_1 = \begin{bmatrix} d_1 & \mathbf{w}_1^T \\ \mathbf{y}_1 & \dot{R}_1 \end{bmatrix}$. The first step of normal Gaussian elimination is to premultiply R_1 by $\begin{bmatrix} 1 & \mathbf{0}^T \\ -\mathbf{y}_1/d_1 & I \end{bmatrix}$, which reduces R_1 to $\begin{bmatrix} d_1 & \mathbf{w}_1^T \\ \mathbf{0} & R_2 \end{bmatrix}$, where $R_2 = \dot{R}_1 - \mathbf{y}_1\mathbf{w}_1^T/d_1$ is the *Schur complement* of d_1 in R_1. At this stage, R_1 has the factorization

$$R_1 = \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{y}_1/d_1 & I \end{bmatrix} \begin{bmatrix} d_1 & \mathbf{w}_1^T \\ \mathbf{0} & R_2 \end{bmatrix} .$$

One then proceeds recursively with the Schur complement $R_2 = \begin{bmatrix} d_2 & \mathbf{w}_2^T \\ \mathbf{y}_2 & \dot{R}_2 \end{bmatrix}$, eventually yielding a factorization $R_1 = LU$, where column k of L is $[\mathbf{0}^T \quad 1 \quad \mathbf{y}_k^T]^T$, and row k of U is $[\mathbf{0}^T \quad 1 \quad \mathbf{w}_k^T]$.

The genesis of *structured* Gaussian elimination is the fact that the displacement structure is preserved under Schur complementation, and that the generators for the Schur complement R_{k+1} can be computed from the generators of R_k in $O(n)$ operations. This is expressed constructively in the following theorem, which is proved in [8].

Theorem 2.1 *Let matrix* $R_1 = \begin{bmatrix} d_1 & \mathbf{w}_1^T \\ \mathbf{y}_1 & \dot{R}_1 \end{bmatrix}$ *satisfy the Sylvester equation*

$$\nabla_{\{A_{f,1},A_{b,1}\}}(R_1) = A_{f,1}R_1 - R_1A_{b,1} = \Phi^{(1)}\Psi^{(1)} , \tag{6}$$

where $\Phi^{(1)} = [\varphi_1^{(1)T} \quad \varphi_2^{(1)T} \quad \cdots \quad \varphi_n^{(1)T}]^T$, $\Psi^{(1)} = [\psi_1^{(1)} \quad \psi_2^{(1)} \quad \cdots \quad \psi_n^{(1)}]$, $\varphi_i^{(1)} \in \mathbf{C}^{1\times\alpha}$ *and* $\psi_i^{(1)} \in \mathbf{C}^{1\times\alpha}$, $(i = 1, 2, \ldots, n)$. *Then* R_2*, the Schur complement of* d_1 *in* R_1*, satisfies the Sylvester equation*

$$\nabla_{\{A_{f,2},A_{b,2}\}}(R_2) = A_{f,2}R_2 - R_2A_{b,2} = \Phi^{(2)}\Psi^{(2)} ,$$

where $A_{f,2}$ *and* $A_{b,2}$ *are respectively* $A_{f,1}$ *and* $A_{b,1}$ *with their first rows and first columns deleted, and where* $\Phi^{(2)} = [0,\ \varphi_2^{(2)T},\ \varphi_3^{(2)T},\ \cdots, \varphi_n^{(2)T}]^T$ *and* $\Psi^{(2)} = [0,\ \psi_2^{(2)},\ \psi_3^{(2)},\ \cdots, \psi_n^{(2)}]$ *are given by*

$$\Phi^{(2)}_{2:n,:} = \Phi^{(1)}_{2:n,:} - \mathbf{y}_1\varphi_1^{(1)}/d_1 , \tag{7}$$

$$\Psi^{(2)}_{:,2:n} = \Psi^{(1)}_{:,2:n} - \psi_1^{(1)}\mathbf{w}_1^T/d_1 . \tag{8}$$

Equations (7) and (8) form the basis of the following general structured Gaussian elimination algorithm.

Algorithm 2.1 (Structured Gaussian elimination)

1. Recover from the generator $\Phi^{(1)}$, $\Psi^{(1)}$ the first row and column of $R_1 = \begin{bmatrix} d_1 & \mathbf{w}_1^T \\ \mathbf{y}_1 & R_{22}^{(1)} \end{bmatrix}$.

2. $[1 \quad \mathbf{y}_1^T/d_1]^T$ and $[d_1 \quad \mathbf{w}_1^T]$ are respectively the first column and row of L_1 and U_1 in the LU factorization of R_1.

3. Compute by equations (7) and (8), the generator $\Phi^{(2)}, \Psi^{(2)}$ for the Schur complement R_2.

4. Proceed recursively with $\Phi^{(2)}$ and $\Psi^{(2)}$. Each major step yields $[1 \quad \mathbf{y}_k^T/d_k]^T$ and $[d_k \quad \mathbf{w}_k^T]$, which are respectively the first column and row of L_k and U_k in the LU factorization of R_k. Column k of L and row k of U are respectively $[\mathbf{0}_{k-1}^T \quad 1 \quad \mathbf{y}_k^T/d_k]^T$ and $[\mathbf{0}_{k-1}^T \quad d_k \quad \mathbf{w}_k^T]$.

Pivoting. Gaussian elimination without pivoting is unstable in general. One normally uses partial pivoting (swapping rows to bring the largest element in the first column to the pivot position) or complete pivoting (swapping rows and columns to bring the largest element in the whole matrix to the pivot position) to improve the accuracy. Row and/or column interchanges can destroy the structure of certain matrices, such as Toeplitz matrices. However, if $A_{f,1}$ in (6) is diagonal (which is the case for Cauchy and Vandermonde type matrices), then the structure is preserved under row permutations.

Partial pivoting can also be incorporated into structured Gaussian elimination. Suppose we wish to swap rows 1 and q of R_1. Let P_1 be the matrix which applies this permutation. Then it is easy to see that P_1R_1 satisfies (6) with the $(1,1)$ and (q,q) entries of $A_{f,1}$ swapped, and with swapped row vectors $\varphi_1^{(1)}$ and $\varphi_q^{(1)}$. Thus, pivoting can be incorporated into Algorithm 2.1 by adding the following steps:

0.5 *Initialization step.* Set permutation matrix $P = I$.

2.5 *After step 2 of Algorithm 2.1.* Let $(\mathbf{y}_1)_q$ be the largest entry by magnitude in $\mathbf{y}_1$. Swap rows 1 and q of P and $\Phi^{(1)}$, and the first and q-th diagonal entries in $A_{f,1}$. Recover the first row of P_1R_1 from $\Psi^{(1)}$ and the swapped $\Phi^{(1)}$.

Note. The computation of the first row of the original $R^{(1)}$ in step 1 of Algorithm 2.1 may be omitted – we only require the first row of the *swapped* $R^{(1)}$.

It may be seen that the pivoted algorithm computes upper and lower triangular matrices L and U which satisfy

$$R^{(1)} = P^T LU \; .$$

Note that for Cauchy-type matrices, where both $A_{f,1}$ and $A_{b,1}$ are diagonal, both row and column pivoting may be performed. However, complete pivoting requires the computation of all the entries in the matrix, which would require $O(n^2)$ operations at each step and $O(n^3)$ operations in all. It will be seen in §6 that a restricted version of row-column pivoting can be used to improve the performance of the GKO algorithm.

2.3 The Cauchy variant of the GKO algorithm (GKO-Cauchy)

Recall that a Cauchy-type matrix satisfies the Sylvester equation (6) with

$$A_{f,1} = D_t = \mathrm{diag}(t_1, t_2, \ldots, t_n) \quad \text{and} \quad A_{b,1} = D_s = \mathrm{diag}(s_1, s_2, \ldots, s_n) \; .$$

It can be easily verified that if $t_i \neq s_j$, then the (i,j) entry of $R^{(1)} = R$ is given by

$$r_{ij} = \frac{\varphi_i \psi_j}{t_i - s_j} \; .$$

There may be some cases where $t_i = s_j$ and $\varphi_i\psi_j = 0$ for some (i,j), and r_{ij} cannot be recovered from its generator. We do not consider these cases in this paper.

In general, at major step k, the reduced matrix $R^{(k)}$ has the form

$$R^{(k)} = \begin{bmatrix} d_1 & & & \mathbf{w}_1^T \\ \mathbf{0} & \ddots & & \vdots \\ \vdots & & d_{k-1} & \mathbf{w}_{k-1}^T \\ \mathbf{0} & \cdots & \mathbf{0} & R_k \end{bmatrix} .$$

The entries of the k-th Schur component R_k, may be computed by

$$\begin{aligned} r_{ij}^{(k)} &= \frac{\varphi_i^{(k)} \psi_j^{(k)}}{t_i - s_j} \, , \quad k \leq i, j \leq n \\ &= (R_k)_{i-k+1, j-k+1} \end{aligned} \tag{9}$$

Equation (9) can be used in Algorithm 2.1 with pivoting to yield the Cauchy version of the GKO algorithm.

Algorithm 2.2 (GKO-Cauchy)

Input. Cauchy-type matrix R_1, specified by $\mathbf{t}$, $\mathbf{s}$, $\Phi^{(1)}$ and $\Psi^{(1)}$.
Output. Factorization $R_1 = P^T LU$, where P is a permutation, and L and U are lower and upper-triangular respectively.

```
% Initialization
$L \leftarrow 0$;   $U \leftarrow 0$;   $P \leftarrow I$
for $k \leftarrow 1 : n$                          % $k$: Iteration number
    for $j \leftarrow k : n$                      % recover col.1 of $R_k$
        $r_{jk}^{(k)} \leftarrow \frac{\varphi_j^{(k)}\psi_k^{(k)}}{t_j - s_k}$
    end
    % Carry out row interchanges
    Find $k \leq q \leq n$ such that $|r_{qk}^{(k)}| = \max_{k \leq j \leq n} |r_{jk}^{(k)}|$
    $t_k \leftrightarrow t_q$;   $\varphi_k^{(k)} \leftrightarrow \varphi_q^{(k)}$;   $r_{kk}^{(k)} \leftrightarrow r_{qk}^{(k)}$
    swap $k$-th and $q$-th rows of $L$
    swap $k$-th and $q$-th rows of $P$

    for $j \leftarrow k+1 : n$                    % Recover row 1 of swapped $R_k$
        $r_{kj}^{(k)} \leftarrow \frac{\varphi_k^{(k)}\psi_j^{(k)}}{t_k - s_j}$
    end
    $u_{kk} \leftarrow r_{kk}^{(k)}$
    % Compute row and col $k$ of $L$ and $U$, and update $\Phi$ and $\Psi$ using (7) and (8)
    for $j \leftarrow k+1 : n$
        $l_{jk} \leftarrow r_{jk}^{(k)} / r_{kk}^{(k)}$
        $u_{kj} \leftarrow r_{kj}^{(k)}$
        $\psi_j^{(k+1)} \leftarrow \psi_j^{(k)} - \psi_k^{(k)} u_{kj} / u_{kk}$
        $\varphi_j^{(k+1)} \leftarrow \varphi_j^{(k)} - \varphi_k^{(k)} l_{jk}$
    end
end
```

2.4 The Toeplitz variant of the GKO algorithm (GKO-Toeplitz)

Recall that a Toeplitz matrix satisfies the Sylvester equation (1), with A_f, A_b, Φ and Ψ being given by equations (3) to (5), and a Toeplitz-type matrix is one with A_f and A_b given by (3), and with general low-rank Φ and Ψ. The first row and column of the Toeplitz-type matrix can be simply generated from Φ and Ψ, and this generating formula can be used in Algorithm 2.1 to yield a structured Gaussian elimination algorithm for Toeplitz-type matrices.

Because neither A_f nor A_b is diagonal, pivoting cannot be introduced directly into this structured algorithm – pivoting will destroy the Toeplitz-type property. However, the Toeplitz-type matrix can be easily converted, by fast orthogonal transformations, into a Cauchy-type matrix which can be factorized as in Algorithm 2.1. The inverse orthogonal transforms yield the factorization of the original matrix. The following result of [8] shows how this conversion may be done.

Theorem 2.2 *Let T be a Toeplitz-type matrix, satisfying*

$$\nabla_{\{Z_1,Z_{-1}\}}(T) = \Omega\Gamma \ ,$$

$$\Omega = [\omega_1^T \quad \omega_2^T \quad \cdots \omega_n^T]^T, \quad \Gamma = [\gamma_1 \quad \gamma_2 \quad \cdots \quad \gamma_n],$$

where the $\{\omega_i\}$ and the $\{\gamma_i\}$ are $1 \times \alpha$ and $\alpha \times 1$ respectively.
Then

$$R = FTD^{-1}F^* \tag{10}$$

is a Cauchy-type matrix, satisfying

$$\nabla_{\{D_F,D_{F_-}\}} = \Phi\Psi \ ,$$

where $F = \frac{1}{\sqrt{n}}[e^{2\pi i(k-1)(j-1)/n}]_{1\leq k,j\leq n}$ is the Discrete Fourier Transform matrix,

$$D_F = \mathrm{diag}(1, e^{2\pi i/n}, \ldots, e^{2\pi i(n-1)/n}) \ , \quad D_{F_-} = \mathrm{diag}(e^{\pi i/n}, e^{3\pi i/n}, \ldots, e^{\pi i(2n-1)/n}) \ , \tag{11}$$

$$D = \mathrm{diag}(1, e^{\pi i/n}, \ldots, e^{\pi i(n-1)/n})$$

and

$$\Phi = F\Omega \ , \qquad \Psi^* = FD\Gamma^* \ . \tag{12}$$

Theorem 2.2 allows the generators of T to be converted to the generators of R in $O(2\alpha n \log n)$ operations via FFTs. R can then be factorized as $R = P^TLU$ using Algorithm 2.2. Using (10), we then obtain

$$T = F^*P^TLUFD \ . \tag{13}$$

From this factorization, a linear system in T can be solved in $n^2 + 2n \log n$ operations, so the whole procedure of conversion of Cauchy form, factorization and solution requires $O(n^2)$ operations.

3 Error Analysis of the GKO-Cauchy Algorithm

In this section, a backward error analysis will be carried out, which yields a bound for the perturbation matrix E, defined by

$$\tilde{L}\tilde{U} = R + E \ , \tag{14}$$

where R is the matrix to be factorized, and $\tilde{L}$ and $\tilde{U}$ are the *computed* factors. In the analysis, we first derive some preliminary results which apply to *any* algorithm for structured Gaussian elimination (SGE), and indicate a general methodology for error analysis of SGE algorithms. We then carry out the analysis for Cauchy-type matrices in general and for the Cauchy-type matrix derived from a Toeplitz matrix by equation (10).

3.1 Preliminary results

The following two lemmas may be used for the error analysis of SGE algorithms in general, and the GKO-Cauchy algorithm in particular. The first lemma shows that if G is the perturbation in the Sylvester equation caused by replacing R by $\tilde{L}\tilde{U}$, then the displacement of E is G.

Lemma 3.1 *Let R be a general structured matrix that satisfies* (1)*, let A_f, A_b, Φ and Ψ be as defined above, and let $\tilde{L}$, $\tilde{U}$ and E be as in* (14)*. Suppose $\tilde{L}$ and $\tilde{U}$ satisfy*

$$A_f\tilde{L}\tilde{U} - \tilde{L}\tilde{U}A_b = \Psi\Phi + G \; ; \tag{15}$$

then E satisfies

$$\nabla_{\{A_f,A_b\}}(E) \equiv A_fE - EA_b = G \; . \tag{16}$$

Proof. From (14) and (15),

$$A_f(R+E) - (R+E)A_b = \Phi\Psi + G \; .$$

Expanding the above, and using (1) we obtain (16). □

Corollary 3.2 *If R is a Cauchy-type matrix with $A_f = D_t$ and $A_b = D_s$, then E satisfies*

$$D_tE - ED_s = G \tag{17}$$

and

$$e_{ij} = \frac{g_{ij}}{t_i - s_j} \; , \quad i,j = 1,\ldots,n \tag{18}$$

Proof. (17) follows directly from (16), and (18) follows by evaluating each component of (17). □

If R is a Toeplitz-type matrix, $A_f = Z_1$, $A_b = Z_{-1}$, and E satisfies $Z_1E - EZ_{-1} = G$. Because Z_1 and Z_{-1} are not diagonal, the recovery formula for E is a little more involved, and will be derived in the next section (Lemma 4.3).

The second lemma of this section shows that G is the sum of the local perturbation matrices incurred in each step of the relevant structured Gaussian elimination (SGE) algorithm.

Lemma 3.3 *Let $\nabla_{\{A_f,A_b\}}$ be the displacement operator as defined in* (1)*; let $\tilde{L}$, $\tilde{U}$ and G be as defined above; let the $\{\tilde{\Phi}^{(k)}, \tilde{\Psi}^{(k)}\}_{k=1,2,\ldots}$ be the computed generators of the $\{R'_k\}_{k=1,2,\ldots}$, the reduced matrices at step k of SGE, and define $\tilde{\Phi}^{(n+1)} = \tilde{\Psi}^{(n+1)} = \mathbf{0}$. Then*

$$G = \sum_{k=1}^{n} H_k \; , \tag{19}$$

where H_k, the local perturbation in each step of SGE, is defined by

$$\nabla_{\{A_f,A_b\}}(\tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:}) = \tilde{\Phi}^{(k)}\tilde{\Psi}^{(k)} - \tilde{\Phi}^{(k+1)}\tilde{\Psi}^{(k+1)} + H_k \; , \quad k = 1,\ldots,n \; . \tag{20}$$

Proof. Writing (20) explicitly, we get

$$A_f\tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:} - \tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:}A_b = \tilde{\Phi}^{(k)}\tilde{\Psi}^{(k)} - \tilde{\Phi}^{(k+1)}\tilde{\Psi}^{(k+1)} + H_k \; , \quad k = 1,\ldots,n \; . \tag{21}$$

Summing the members of (21), we obtain

$$A_f\sum_{i=1}^{n}\tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:} - \sum_{i=1}^{n}\tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:}A_b = \tilde{\Phi}^{(1)}\tilde{\Psi}^{(1)} - \tilde{\Phi}^{(n+1)}\tilde{\Psi}^{(n+1)} + \sum_{i=1}^{n} H_k \; . \tag{22}$$

Now $\sum_{i=1}^{n}\tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:} = \tilde{L}\tilde{U}$, $\tilde{\Phi}^{(1)} = \Phi$, $\tilde{\Psi}^{(1)} = \Psi$ and $\tilde{\Phi}^{(n+1)} \equiv \tilde{\Psi}^{(n+1)} \equiv \mathbf{0}$. Substituting these identities into (22) and comparing the resulting relation with (15), we obtain (19) □

3.2 Methodology of error analysis for SGE algorithms

Lemmas 3.1 and 3.3 may be used in a general methodology for the error analysis of SGE algorithms similar to Algorithm 2.1.

In the following methodology and the subsequent analysis of the GKO algorithm, we now let $\Phi^{(k)}$ and $\Psi^{(k)}$ be the *computed* values of these quantities, $\mathbf{u}_{k:}$, $\mathbf{r}^{(k)}_{k:n,k}$, $\mathbf{l}_{:k}$, $\Phi^{(k+1)}$ and $\Psi^{(k+1)}$ be the values of these quantities computed in exact arithmetic from $\Phi^{(k)}$ and $\Psi^{(k)}$ using steps 1 to 3 of Algorithm 2.1, and $\tilde{\mathbf{u}}_{k:}$, $\tilde{\mathbf{r}}^{(k)}_{k:n,k}$, $\tilde{\mathbf{l}}_{:k}$, $\tilde{\Phi}^{(k+1)}$ and $\tilde{\Psi}^{(k+1)}$ be the actual computed values of $\mathbf{u}_{k:}$, $\mathbf{r}^{(k)}_{k:n,k}$, $\mathbf{l}_{:k}$, $\Phi^{(k+1)}$ and $\Psi^{(k+1)}$ respectively. The methodology is as follows:

1. Using a standard rounding error analysis, derive expressions of the form

$$\begin{aligned}
\tilde{\mathbf{u}}_{k:} &= \mathbf{u}_{k:} + \delta\tilde{\mathbf{u}}_{k:} && (23)\\
\tilde{\mathbf{r}}^{(k)}_{k:n,k} &= \mathbf{r}^{(k)}_{k:n,k} + \delta\tilde{\mathbf{r}}^{(k)}_{k:n,k} && (24)\\
\tilde{\mathbf{l}}_{:k} &= \mathbf{l}_{:k} + \delta\tilde{\mathbf{l}}_{:k} && (25)\\
\tilde{\Phi}^{(k+1)} &= \Phi^{(k)} - \tilde{\mathbf{l}}_{:k}\phi^{(k)}_k + \delta\tilde{\Phi}^{(k+1)} && (26)\\
\tilde{\Psi}^{(k+1)} &= \Psi^{(k)} - \psi^{(k)}_k\tilde{\mathbf{u}}_{k:}/\tilde{r}^{(k)}_{kk} + \delta\tilde{\Psi}^{(k+1)} && (27)
\end{aligned}$$

   where $\delta\tilde{\mathbf{u}}_{k:}$, etc. are error terms.

2. Evaluate $\Phi^{(k)}\Psi^{(k)} - \tilde{\Phi}^{(k+1)}\tilde{\Psi}^{(k+1)}$ using (23) to (27). This can be expressed in the form

$$\Phi^{(k)}\Psi^{(k)} - \tilde{\Phi}^{(k+1)}\tilde{\Psi}^{(k+1)} = A_f\tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:} - \tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:}A_b + F_k \ , \tag{28}$$

   where F_k is an error term. By (21),

$$H_k = -F_k \ .$$

3. After some manipulation, express F_k as a sum of terms of the form

$$S(A_f, A_b) \circ T(V^{(k)}) \circ \tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:} \circ \hat{\Delta} \quad \text{or} \quad S(A_f, A_b) \circ T(V^{(k+1)}) \circ L_{:,k+1:n}U_{k+1:n,:} \circ \hat{\Delta} \ .$$

   Here, the $S(A_f, A_b)$ are matrices formed from A_f and A_b, Δ is a matrix whose elements are bounded in magnitude by ϵ, and $V^{(k)}$ is defined by

$$|\Phi^{(k)}||\Psi^{(k)}| \equiv V^{(k)} \circ \Phi^{(k)}\Psi^{(k)} \ . \tag{29}$$

4. Apply (19) to derive an expression for G.

5. Lemma 3.1 shows that G satisfies

$$\nabla_{\{A_f,A_b\}}E = G \ . \tag{30}$$

   Using the appropriate algorithm to recover a structured matrix from its generators, derive an expression for E from the expression for G. Note that in general, G will be of full rank. However, (30) will still be satisfied by E and G.

6. Derive bounds for $\|E\|$ using some norm.

3.3 Error analysis of GKO for Cauchy-type matrices

In this subsection, we use the above methodology to derive the first of our main results — a bound for $\|E\|$ when a Cauchy matrix R is factorized by the GKO algorithm. The results are encapsulated in three theorems, which yield expressions for the $\{H_k\}$, an elementwise bound for G, and a bound for $\|E\|$ respectively. We then discuss the size of the bound for $\|E\|$.

Theorem 3.4 *Let R be a Cauchy matrix to be factorized by the GKO algorithm and let F_k, H_k, $V^{(k)}$, $\tilde{\mathbf{l}}_{:k}$, $\tilde{\mathbf{u}}_{k:}$ be as defined above. Then*

$$\begin{aligned} F_k &= c_1\hat{\Delta}^{(1)} \circ D_{vc}^{(k)} D_p \tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:} + c_2\tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:} D_q D_{vr}^{(k)} \circ \hat{\Delta}^{(2)} + c_3(r_{kk}^{(k)})^{-1} v_{kk}^{(k)} \hat{\Delta}^{(3)} \circ \tilde{\mathbf{l}}_{:k}\tilde{\mathbf{u}}_{k:} + \\ &\quad c_4\hat{\Delta}^{(4)} \circ B^I \circ V^{(k+1)} \circ \tilde{L}_{:,k+1:n}\tilde{U}_{k+1:n,:} \end{aligned}$$

where c_1 to c_4 are small constants, $D_{vc}^{(k)} = \mathrm{diag}(v_{:k}^{(k)})$, $D_{vr}^{(k)} = \mathrm{diag}(v_{k:}^{(k)})$, $D_p = \mathrm{diag}\{t_i - s_k\}_i$, $D_q = \mathrm{diag}\{t_k - s_j\}_j$, $B = [1/(t_i - s_j)]$ is the ordinary Cauchy matrix with displacement operator $\nabla_{\{D_s,D_t\}}$, the $\hat{\Delta}^{(\cdot)}$ are matrices whose elements are less than ϵ in magnitude, and $H_k = -F_k$.

Proof. In the following, we simplify our notation and drop the superscript (k); where the superscript is $(k+1)$ we indicate this by a prime $(')$; and we drop the subscripts $:k$, $k:$ and $k:n,k$. In the following, we do not give all the steps in the derivation of the various expressions, as these are straightforward but very tedious. However, we indicate how key intermediate expressions are derived.

We use the normal properties of floating point operations performed with at least one guard digit, viz. $fl(a) = a(1+\delta_1)$ and $fl(a\star b) = (a\star b)(1+\delta_2)$, where $fl(a)$ denotes rounding, $fl(a\star b)$ is the computed result of any of the four basic floating-point operations, and $|\delta_1|, |\delta_2| < \epsilon$.

Following step 1 of the above methodology, we evaluate expressions for the computed values of $\tilde{\mathbf{r}}$, $\tilde{\mathbf{l}}$ and $\tilde{\mathbf{u}}$ (subscripts and superscripts dropped), yielding after a few steps

$$\tilde{\mathbf{u}} = \mathbf{u} + 2\tilde{\mathbf{u}}\Delta^{(1)} + \phi_k \mathcal{D}^{(1)}\Psi D_q^{-1} \; , \tag{31}$$

$$\tilde{\mathbf{r}} = \mathbf{r} + 2\Delta^{(2)}\tilde{\mathbf{r}} + D_p^{-1}\mathcal{D}^{(2)}\Phi\psi_k \; , \tag{32}$$

$$\tilde{\mathbf{l}} = \mathbf{l} + 5\Delta^{(3)}\tilde{\mathbf{l}} + \tilde{r}_{kk}^{-1} D_p^{-1}\mathcal{D}^{(2)}\Phi\psi_k - b_{kk}\tilde{r}_{kk}^{-1}\partial_k^{(2)}\phi_k\psi_k\tilde{\mathbf{l}} \; .$$

Here and below the $\Delta^{(\cdot)}$ denote diagonal matrices with elements of magnitude less than ϵ; the $\mathcal{D}^{(\cdot)}$ are elementwise operators which multiply each element of their matrix operands by a factor less than ϵ, and the $\partial_k^{(\cdot)}$ are similar elementwise vector operators.

Similarly, it can be shown that the computed values of Φ' and Ψ' satisfy

$$\begin{aligned} \tilde{\Phi}' &= \Phi - \tilde{\mathbf{l}}\phi_k + \mathcal{D}^{(3)}\Phi' + \mathcal{D}^{(4)}(\tilde{\mathbf{l}}\phi_k) \; , \\ \tilde{\Psi}' &= \Psi - \psi_k\tilde{\mathbf{u}}/\tilde{r}_{kk} + \mathcal{D}^{(5)}\Psi' + 2\mathcal{D}^{(6)}(\psi_k\tilde{\mathbf{u}})/\tilde{r}_{kk} \; . \end{aligned}$$

Carrying out step 2 of the above methodology, we obtain

$$\begin{aligned} \Phi\Psi - \tilde{\Phi}'\tilde{\Psi}' &= \Phi\psi_k\tilde{\mathbf{u}}/\tilde{r}_{kk} + \tilde{\mathbf{l}}\phi_k\Psi - \tilde{\mathbf{l}}\phi_k\psi_k\tilde{\mathbf{u}}/\tilde{r}_{kk} - 2\Phi\mathcal{D}^{(6)}(\psi_k\tilde{\mathbf{u}})/\tilde{r}_{kk} - \Phi'\mathcal{D}^{(5)}\Psi' - \\ &\quad \mathcal{D}^{(4)}(\tilde{\mathbf{l}}\phi_k)\Psi - \mathcal{D}^{(3)}\Phi'\Psi' + \mathcal{D}^{(4)}(\tilde{\mathbf{l}}\phi_k)\psi_k\tilde{\mathbf{u}}/\tilde{r}_{kk} + 2\tilde{\mathbf{l}}\phi_k\mathcal{D}^{(6)}(\psi_k\tilde{\mathbf{u}})/\tilde{r}_{kk} \; . \end{aligned} \tag{33}$$

Let T_3 denote the first three terms in (33). From Algorithm 2.2, we have $\Phi\psi_k = D_p\mathbf{r}$ and $\phi_k\Psi = \tilde{\mathbf{u}}D_q$. Using these relations in T_3, and expressing $\mathbf{r}$ in terms of ($\tilde{\mathbf{r}}$ - error terms) using (32) and $\mathbf{u}$ in terms of ($\tilde{\mathbf{u}}$ - error terms) using (31), we can show that

$$\begin{aligned} T_3 &= D_t\tilde{\mathbf{l}}\tilde{\mathbf{u}} - \tilde{\mathbf{l}}\tilde{\mathbf{u}}D_s - 3D_p\Delta^{(4)}\tilde{\mathbf{l}}\tilde{\mathbf{u}} - 2\tilde{\mathbf{l}}\tilde{\mathbf{u}}\Delta^{(5)}D_q + 2r_{kk}^{-1}\delta\tilde{\mathbf{l}}\tilde{\mathbf{u}} - \\ &\quad \mathcal{D}^{(2)}\Phi\psi_k\tilde{\mathbf{u}}/r_{kk} - \tilde{\mathbf{l}}\phi_k\mathcal{D}^{(1)}\Psi + \tilde{r}_{kk}^{-1}\partial_k^{(2)}\phi_k\psi_k\tilde{\mathbf{l}}\tilde{\mathbf{u}} \; , \end{aligned} \tag{34}$$

where $|\delta| < \epsilon$. By using (34) for the first three terms of (33), we get an equation of the form (28), where F_k is given by the last six terms in (33) plus the last six terms in (34). Terms involving the $\mathcal{D}^{(\cdot)}$ may be expressed in terms of $\tilde{\mathbf{l}}\tilde{\mathbf{u}}$ or $\tilde{L}\tilde{U}$ by using the definition of V, which in the current notation is

$$v_{ij} = \frac{|\phi_i||\psi_j|}{\phi_i \psi_j} \; .$$

Consider the factor $\Phi\mathcal{D}^{(6)}(\psi_k\tilde{\mathbf{u}})/\tilde{r}_{kk}$ in the term $-2\Phi\mathcal{D}^{(6)}(\psi_k\tilde{\mathbf{u}})/\tilde{r}_{kk}$. We have

$$(\Phi\mathcal{D}^{(6)}(\psi_k\tilde{\mathbf{u}})/\tilde{r}_{kk})_{ij} = \phi_i\partial_j^{(6)}(\psi_k\tilde{u}_j)/\tilde{r}_{kk} \; .$$

Recall that $\phi_i = [\phi_{i1}, \phi_{i2}]$ and $\psi_j = [\psi_{1j}, \psi_{2j}]$. Then

$$(\Phi\mathcal{D}^{(6)}(\psi_k\tilde{\mathbf{u}})/\tilde{r}_{kk})_{ij} = (\phi_{i1}\delta_{1j}^{(6)}\psi_{1k} + \phi_{i2}\delta_{2j}^{(6)}\psi_{2k})\tilde{u}_j/\tilde{r}_{kk} \; ,$$

where $\delta_{1j}^{(6)}$ and $\delta_{1j}^{(6)}$ are the scaling factors from the operator $\partial_j^{(6)}$. From the definition of V, using the fact that $\tilde{l}_i \doteq \tilde{r}_{ik}/\tilde{r}_{kk}$, this can be shown to be

$$(\Phi\mathcal{D}^{(6)}(\psi_k\tilde{\mathbf{u}})/\tilde{r}_{kk})_{ij} = \hat{\delta}_{ij}^{(6)} v_{ik} b_{ik}^{-1} \tilde{l}_i \tilde{u}_j \; ,$$

where $|\hat{\delta}_{ij}^{(6)}| \leq \max_{j=1,2} |\delta_{kj}^{(6)}|$. In matrix form, we obtain

$$\Phi\mathcal{D}^{(6)}(\phi_k\tilde{\mathbf{u}})/\tilde{r}_{kk} = \hat{\Delta} \circ \operatorname{diag}\{v_{ik}/b_{ik}\}\tilde{\mathbf{l}}\tilde{\mathbf{u}} \; ,$$

where $\hat{\Delta}$ and subsequent $\hat{\Delta}^{(\cdot)}$ are matrices with elements bounded in magnitude by ϵ. Similarly, all the other terms can be expressed as either

(i) an elementwise product of $\hat{\Delta}^{(\cdot)}$ and a normal product of $\tilde{\mathbf{l}}\tilde{\mathbf{u}}$ and matrices derived from B or V, or

(ii) an elementwise product of the form $\hat{\Delta}^{(\cdot)} \circ B^I \circ V' \circ L_{:,k+1:n} U_{k+1:n,:}$.

When this is done, the result follows. □

The next theorem uses Lemma 3.3 to obtain an elementwise bound for $|G|$.

Theorem 3.5 *Let H_k be as in Theorem 3.4. Then*

$$\begin{aligned} |G| \;&\leq\; c_1 b_{\min}^{-1}\hat{\Delta}^{(1)} \circ |\hat{L}||U| + c_2 b_{\min}^{-1}|L||\hat{U}| \circ \hat{\Delta}^{(2)} + c_3 b_{\min}^{-1}\hat{\Delta}^{(3)} \circ |L|\operatorname{diag}\{v_{kk}^{(k)}\}|U| \\ &+\; c_4 |B^I| \circ \hat{\Delta}^{(4)} \circ \sum_{k=2}^{n} |\hat{R}_k'| \end{aligned}$$

where $b_{\min}$ is the minimum modulus of the elements of B, $\hat{L} = [\mathbf{v}_{:k}^{(k)}]_{k=1}^n \circ L$, $\hat{U} = U \circ [\mathbf{v}_{k:}^{(k)}]_{k=1}^n$, and $\hat{R}_k' = V^{(k)} \circ L_{:,k:n} U_{k:n,:}$.

Proof. G is evaluated by carrying out the summation in (19), and using the identities $\sum_{i=k}^n \mathbf{a}_{:k}\mathbf{b}_{k:} = AB$ and $\sum_{i=k}^n x_k \mathbf{a}_{:k}\mathbf{b}_{k:} = A\operatorname{diag}\{x_k\}B$. □

We now apply the last step in the above methodology to derive an expression for $\|E\|$.

Theorem 3.6 *Let E be the backward error $E = \tilde{L}\tilde{U} - R$ in the factorization of R using the GKO algorithm, let $\hat{L}$, $\hat{U}$, $\hat{R}$, B and V be as above. Then $\|E\|$ is bounded by*

$$\|E\| \leq \epsilon \left(c_5 \frac{b_{\max}}{b_{\min}} g_1 + c_6 n g_2 \right) \|L\| \|U\| \; , \tag{35}$$

where the Frobenius norm is used, $b_{\max}$ and $b_{\min}$ are the maximum and minimum moduli of the elements of B, c_5 and c_6 are small constants, and g_1 and g_2 are generator growth factors, *defined by*

$$\begin{aligned} g_1 &= c_7 \frac{\|\hat{L}\|}{\|L\|} + c_8 \frac{\|\hat{U}\|}{\|U\|} + c_9 \|\mathrm{diag}\{v_{kk}^{(k)}\}\| \; , & (36) \\ g_2 &= \max_{k=2,\ldots,n} \{ |\hat{R}_k\| / \|R_k\| \} \; , & (37) \end{aligned}$$

with $c_7, c_8, c_9 < 1$.

Proof. From step 5 of the above methodology, we essentially invert the Sylvester equation (30) to derive an expression for E. To do this we apply (18) in Corollary 3.2. This can be written in matrix form

$$E = B \circ G$$

so

$$\begin{aligned} |E| = |B| \circ |G| \;\; \leq \;\; & c_1 \frac{b_{\max}}{b_{\min}} \hat{\Delta}^{(5)} \circ |\hat{L}||U| + c_2 \frac{b_{\max}}{b_{\min}} \|L||\hat{U}| \circ \hat{\Delta}^{(6)} + \\ & c_3 \frac{b_{\max}}{b_{\min}} \hat{\Delta}^{(7)} \circ |L| \mathrm{diag}\{v_{kk}\} |U| + c_4 \Delta^{(8)} \circ \sum_{k=2}^{n} |\hat{R}_k'| \; . \end{aligned} \tag{38}$$

We now define $g_2 \equiv \max_{k=2,\ldots,n} \|\hat{R}^{(k)}\| / \|R^{(k)}\|$, $g_4 \equiv \|\hat{L}\| / \|L\|$, $g_5 \equiv \|\hat{U}\| / \|U\|$ and $g_6 \equiv \|\mathrm{diag}\{v_{kk}^{(k)}\}\|$. These can be considered to be generator growth factors — they are functions of the $V^{(k)}$, which from the definition (29) are the ratio of the products of the magnitudes of the generators to the products of the generators. We will see in §5 that these growth factors can sometimes be large.

Taking the Frobenius norm of (38), we can easily show that

$$\|E\| \leq c_1 \delta_1 \frac{b_{\max}}{b_{\min}} g_3 \|L\| \|U\| + c_2 \delta_2 g_4 \frac{b_{\max}}{b_{\min}} \; \|L\| \|U\| + c_3 \delta_3 \frac{b_{\max}}{b_{\min}} g_5 \|L\| \|U\| + c_6 n \delta_4 g_2 \|L\| \|U\| \; . \tag{39}$$

where $0 \leq |\delta_1|, \ldots, |\delta_4| < \epsilon$. The result follows by collecting the first three terms of (39). □

The following corollary specializes the above result to the case when R is derived from a Toeplitz matrix.

Corollary 3.7 *Let R be derived from a Toeplitz matrix T by the transformation* (10) *in Theorem 2.2, and let c_1, c_2, g_1, g_2 and E be as defined in Theorem 3.6. Then $\|E\|$ is bounded by*

$$\|E\| \leq \epsilon c_{10} g_3 n \|L\| \|U\| \; , \tag{40}$$

where $c_{10} = \max(2c_5/\pi, c_6)$ and $g_3 = \max(g_1, g_2)$.

Proof. Recall that $B = [1/(t_i - s_j)]$ is the ordinary Cauchy matrix with displacement operator $\nabla_{\{D_s, D_t\}}$; from equations (11) in Theorem 2.2, the t_i are n equally-spaced points around the unit circle, including one at (1,0), and the s_j are also n equally-spaced points around the unit circle, with each s_j between two of the t_i. Clearly $\pi/n < t_i - s_j < 2 \;\; \forall i, j$, so by the definition of B,

$$\frac{b_{\max}}{b_{\min}} < 2n/\pi \; . \tag{41}$$

Substituting (41) in (39), bounding $2c_5/\pi$ and c_6 by c_{10}, and bounding g_1 and g_2 by g_3 yields the result. □

The above results show that the expressions for the backward error bounds from the GKO algorithm are similar to the ones for Gaussian elimination with partial pivoting (GE/PP) [9], except for the generator growth factors which might arise in particular cases where the $\Phi^{(k)}$ and $\Psi^{(k)}$ are large, but not the $\Phi^{(k)}\Psi^{(k)}$ or the R_k. So there may be some cases where large error growth may occur in the GKO algorithm but not GE/PP. In §5, we give an example where this occurs.

4 Error Analysis of the GKO-Toeplitz Algorithm

Recall that the steps in the GKO-Toeplitz algorithm are (i) compute the generators from the Toeplitz matrix T using (4) and (5), (ii) convert them to generators of a Cauchy matrix using (12) and (iii) compute factors L and U of this Cauchy matrix using the GKO algorithm. The factors of T are then given by (13). There are errors incurred at each of these steps. In this section, we do not consider permutations, as these do not contribute to the error. We will derive a bound for the perturbation matrix E_T, defined by

$$F^* \tilde{L} \tilde{U} F D = T + E_T \; .$$

In our development, we show in Theorem 4.1 that E_T consists of two components — the first due to the error $\|E\|$ incurred in the Cauchy factorization and the second due to the errors incurred in computing the Cauchy generators $\tilde{\Phi}$ and $\tilde{\Psi}$. The latter is a Toeplitz-type perturbation ΔT such that $T + \Delta T$ transforms exactly to $\tilde{\Phi}$ and $\tilde{\Psi}$. We then derive two lemmas needed to derive ΔT, and then present the main result of this section in Theorem 4.4.

4.1 Main components of E_T

E_T has two main components, as is shown in the following.

Theorem 4.1 *Let F and D be as in Theorem 2.2, let $\tilde{\Phi}$ and $\tilde{\Psi}$ be the Cauchy generators computed using* (4), (5) *and* (12), *and let $\tilde{L}$ and $\tilde{U}$ be the factors computed from $\tilde{\Phi}$ and $\tilde{\Psi}$ using the GKO algorithm. Then the perturbed factorization of T satisfies*

$$F^* \tilde{L} \tilde{U} F D \equiv T + E_T = T - F^* E F D + \Delta T \; , \tag{42}$$

where E is as in Theorem 3.6 and ΔT is a Toeplitz-type perturbation of T such that $T + \Delta T$ has generators $\tilde{\Omega}$ and $\tilde{\Gamma}$ that transform exactly to $\tilde{\Phi}$ and $\tilde{\Psi}$ using (4), (5) *and* (12).

Proof. Let $\tilde{R}$ be the Cauchy matrix generated by $\tilde{\Phi}$ and $\tilde{\Psi}$. We have

$$\tilde{R} = \tilde{L}\tilde{U} + E \; ,$$

and we know from (10) that $\tilde{\Phi}$ and $\tilde{\Psi}$ are the generators for

$$\tilde{R} = F(T + \Delta T) D^{-1} F^*$$

where $T + \Delta T$ is some Toeplitz-type matrix. From the above two equations we obtain

$$T + \Delta T = F^* \tilde{R} F D = F^* (\tilde{L}\tilde{U} + E) F D \; ,$$

from which the desired result follows. □

Thus, by (42), we see that E_T has one component with the same norm bound as E, and another which perturbs T to a matrix such that its generators, say $\tilde{\Omega}$ and $\tilde{\Gamma}$, transform exactly to $\tilde{\Phi}$ and $\tilde{\Psi}$. Before we derive an expression for ΔT, we need two preliminary results : expressions for $\tilde{\Omega}$ and $\tilde{\Gamma}$, and a method to recover $T + \Delta T$ from its generators $\tilde{\Omega}$ and $\tilde{\Gamma}$.

4.2 Estimation of ΔT — preliminary results

The required results are given in the following two lemmas.

Lemma 4.2 *Let Ω and Γ be as in* (4) *and* (5)*, and let $\tilde{\Omega}$ and $\tilde{\Gamma}$ transform exactly to $\tilde{\Phi}$ and $\tilde{\Psi}$ using* (12)*. Let $[\mathbf{a}, \mathbf{b}] = \tilde{\Omega} - \Omega$ and let $[\mathbf{c}, \mathbf{d}\,] = \tilde{\Gamma}^* - \Gamma^*$. Then*

$$\mathbf{a} = \mathbf{0}$$

and $\|\mathbf{b}\|$, $\|\mathbf{c}\|$ and $\|\mathbf{d}\|$ are bounded by

$$\begin{aligned} \|\mathbf{b}\| &\leq \epsilon k_1 n^{3/2} \|\boldsymbol{\omega}_{:2}\| \, , & (43) \\ \|\mathbf{c}\| &\leq \epsilon k_2 n^{3/2} \|\boldsymbol{\gamma}_{1:}\| \, , & (44) \\ \|\mathbf{d}\| &\leq \epsilon \, . & (45) \end{aligned}$$

Proof. We first consider the errors incurred in the computation of $\tilde{\Phi}$ and $\tilde{\Psi}$. We have

$$\begin{aligned} \tilde{\Phi} &= fl\{\tilde{F}[\mathbf{e}_1, \tilde{\boldsymbol{\omega}}_{:2}]\}, \quad \text{where } \tilde{F} = fl(F)\ , \tilde{\boldsymbol{\omega}}_{:2} = fl(\boldsymbol{\omega}_{:2}) \\ &= [\mathbf{1}, fl(\tilde{F}\tilde{\boldsymbol{\omega}}_{:2})]\ , \quad \text{where } \mathbf{1} = [1, 1, \ldots, 1]^T \\ &= [\mathbf{1}, \tilde{F}\tilde{\boldsymbol{\omega}}_{:2} + k_3 n \|\tilde{\boldsymbol{\omega}}_{:2}\| \boldsymbol{\delta}^{(1)}] \end{aligned}$$

where $|\delta_i^{(1)}| < \epsilon\, , i = 1, \ldots, n$. After a few more steps, this becomes

$$\tilde{\Phi} = F[\mathbf{e}_1, \boldsymbol{\omega}_{:2} + \mathbf{b}]$$

where $\mathbf{b} = \Delta^{(7)}\boldsymbol{\omega}_{:2} + k_4(n+1)\|\tilde{\boldsymbol{\omega}}_{:2}\| F^* \boldsymbol{\delta}^{(1)}$. In a similar way, it can be shown that

$$\tilde{\Psi}^* = FD[\boldsymbol{\gamma}_{1:}^* + \mathbf{c}, \mathbf{e}_n + \mathbf{d}]$$

where $\mathbf{c} = k_5 D^* \Delta^{(8)} D \boldsymbol{\gamma}_{1:}^* + k_6(n+1)\|\boldsymbol{\gamma}_{1:}\| \boldsymbol{\delta}^{(2)}$ and $\mathbf{d} = D^* F^* d_n \Delta^{(9)} \mathbf{f}_{n:}^T$. Now the expressions in square brackets transform exactly to $\tilde{\Omega}$ and $\tilde{\Gamma}$ respectively, and by taking norms of $\mathbf{b}$, $\mathbf{c}$ and $\mathbf{d}$ the bounds (43) to (45) can be demonstrated in a few steps. □

Lemma 4.3 *For any matrix A, let $\nabla_{\{Z_1, Z_{-1}\}} A = B$. Then A can be recovered from B using*

$$a_{ij} = \sum_{k=j}^{n} b_{1+(i+k-j) \bmod n, k} - \sum_{k=1}^{j-1} b_{1+(i+k-j) \bmod n, k}\ . \qquad (46)$$

Proof. From the displacement operator $\nabla_{\{Z_1, Z_{-1}\}}$, the following properties of B are easily seen:

$$\begin{aligned} b_{ij} &= a_{i-1,j} - a_{i,j-1}, \quad 1 < i \leq n\ , 1 \leq j < n\ , & (47) \\ b_{1j} &= a_{nj} - a_{i,j+1}, \quad 1 \leq j < n\ , & (48) \\ b_{in} &= a_{i-1,j} + a_{i1}, \quad 1 < i \leq n \quad \text{and} & (49) \\ b_{1n} &= a_{n,n-1} + a_{11}\ . & (50) \end{aligned}$$

It can be verified that if the elements of A are given by (46), then (47) to (50) are satisfied. □

Equation (46) shows that an element a_{ij} is recovered by computing $x - y$, where x is the sum of elements of B down the diagonal, commencing from $b_{i+1,j}$ and proceeding to the last column, wrapping from the last row to the first if necessary during the summing; y is a similar "wrapped diagonal sum" from the first column to $b_{i,j-1}$.

4.3 Main result

We now use Theorem 4.1, Lemma 4.2 and Lemma 4.3 to derive a bound for the backward error $\|E_T\|$ in the GKO-Toeplitz algorithm.

Theorem 4.4 *Let F and D be as in Theorem 2.2, and let $\tilde{L}$ and $\tilde{U}$ be the factors computed from T using the GKO-Toeplitz algorithm. Then the perturbed factorization of T satisfies*

$$F^*\tilde{L}\tilde{U}FD \equiv T + E_T = T + E^{(1)} + E^{(2)} \ , \tag{51}$$

where $E^{(1)}$ is a general matrix with norm $\|E^{(1)}\| = \|E\|$, E is as in Theorem 3.6, and $E^{(2)}$ is a Toeplitz-type matrix with norm bounded by

$$\|E^{(2)}\| \le \epsilon c_{11} n^2 (\|\mathbf{t}_{1:}\| + \|\mathbf{t}_{:1}\|) \ . \tag{52}$$

Proof. By comparing (51) and (42), we see that $E^{(1)} = -F^*EFD$, and because F and D are orthogonal matrices,

$$\|E^{(1)}\| = \|E\| \ . \tag{53}$$

From the above comparison we also have $E^{(2)} = \Delta T$, a Toeplitz-type perturbation of T such that $T + \Delta T$ has generators $\tilde{\Omega}$ and $\tilde{\Gamma}$ that transform exactly to the Cauchy generators $\tilde{\Phi}$ and $\tilde{\Psi}$ computed using (4). In the following, we use $E^{(2)}$ for ΔT. From Lemma 4.2, we have

$$\nabla(T + E^{(2)}) = \tilde{\Omega}\tilde{\Gamma} = \Omega\Gamma + \mathbf{e}_1\mathbf{c}^* + \boldsymbol{\omega}_{:2}\mathbf{d}^* + \mathbf{b}\mathbf{e}_n^T \ ,$$

where $\mathbf{b}$, $\mathbf{c}$ and $\mathbf{d}$ are bounded as in (43) to (45). The second-order error term $\mathbf{b}\mathbf{d}^*$ has been omitted. We then have

$$\nabla E^{(2)} = \mathbf{e}_1\mathbf{c}^* + \boldsymbol{\omega}_{:2}\mathbf{d}^* + \mathbf{b}\mathbf{e}_n^T \ ,$$

and we use (46) to compute $E^{(2)}$. This yields, after some algebra

$$|\mathbf{e}_{:j}^{(2)}| = C_{j-1}(|\mathbf{c}^R| + |\mathbf{b}^R|) + |\mathbf{p}_{:j}|$$

where C_k is a matrix which by premultiplication, circularly upshifts a vector k places, $\mathbf{x}^R$ indicates the reversal of $\mathbf{x}$, and the moduli of $\mathbf{p}_{:j}$ are bounded by

$$\begin{aligned} |p_{ij}| &\le |\boldsymbol{\omega}_{:2}|^T C_{j-i-1}|\mathbf{d}| \\ &\le \|\boldsymbol{\omega}_{:2}\|\|\mathbf{d}\| \ . \end{aligned} \tag{54}$$

Using (43), (44), (54) and (45) it is easily seen that

$$\|\mathbf{e}_{:j}^{(2)}\| \le c_{12} n^{3/2} (\|\boldsymbol{\omega}_{:2}\| + \|\boldsymbol{\gamma}_{1:}\|) \ .$$

From this, using the definitions (4) and (5), we obtain the bound (52) for $E^{(2)}$. Together with (53), this yields the result. □

5 Discussion of Error Bounds

We first discuss the factors in the above error bounds and relate them to what would be expected for Gaussian elimination with partial pivoting (GE/PP). Then we show, for both the Cauchy and Toeplitz variants, that there are some cases where the backward error growth can be large.

5.1 Relation of bounds to those for GE/PP

Consider the backward error E incurred by the Cauchy variant (equation (35)). The term $\|L\|\|U\|$ is similar to that obtained for GE/PP [9]. However, the first factor contains the generator growth factors g_1 and g_2. These are given by ratios of norms of the hatted quantities to the unhatted quantities in (36) and (37). The former are derived from the latter by element-wise multiplication by submatrices of the $V^{(k)}$, which from their definitions (29) are the ratio of the products of the magnitudes of the generators to the products of the generators. For an ordinary Cauchy matrix, $v_{ij}^{(k)} = 1 \; \forall i, j, k$ because $\Phi^{(k)}$ and $\Psi^{(k)}$ have only one column and row respectively. However, for higher displacement-rank Cauchy matrices, there may be significant cancellation in the computation of the denominator of (29), so they may be significant growth in the size of the $\hat{L}$, $\hat{U}$ and $\hat{R}_k$ compared to the L, U and R_k respectively.

The backward error E_T incurred by the Toeplitz variant has two components — one with the same norm as E above, and a Toeplitz-type component with norm bounded as in (52). The latter bound is proportional to n^2 and contains no growth factors, so it would be expected that the bound would be dominated by the first component.

We next give examples where the generator growth might be expected to be large in the Cauchy and Toeplitz variants.

5.2 Examples of large generator growth

Cauchy case. Here, we can select an example where all the elements of $V = V^{(1)}$ are large. This will occur when significant cancellation occurs in the computation of the $\phi_i \psi_j$. Such an example is

$$\Phi = [\mathbf{a}, \mathbf{a} + \mathbf{f}] \;, \quad \Psi = [\mathbf{a}, -\mathbf{a}]^T \;,$$

where $\|\mathbf{a}\|$ is of order unity, and $\|\mathbf{f}\|$ is very small. Then $\Phi\Psi = -\mathbf{fa}$, that is, all the elements of $\Phi\Psi$ are very small compared to those of $|\Phi||\Psi|$. Moreover, because $\mathbf{a}$ and $\mathbf{f}$ can be arbitrary except for their norms, the original matrix $[(t_i - s_j)^{-1}\phi_i\psi_j]$ is in general well-conditioned.

Toeplitz case. The Toeplitz case has an extra constraint on the selection of Φ and Ψ, since it must be generated from Ω and Γ using the transformations (12). Because of this constraint, there is no case where all the elements of V can be made large. However, all of the first column of V can be made large, and this will cause error growth, in spite of the pivoting. This will happen in the following case.

Recall that $a_{i-j} = t_{ij} \; \forall i, j$. Select

$$\begin{aligned} a_0 &= 1 && (55) \\ \text{and} \quad a_i &= -a_{i-n} \;, \quad 1 \le i \le n-1 \;, && (56) \end{aligned}$$

so that $\Omega = [\mathbf{e}_1, \mathbf{e}_1]$. Then all of the first column of V will be large if $\psi_{11} + \psi_{12}$ is very small compared to ψ_{11} and ψ_{12}. It can be verified from (12) that if we select $a_1, \ldots, a_{n-1}$ to satisfy

$$\sum_{j=1}^{n-1} a_{n-j} \exp(i\pi(j-1)/n) = -\exp(i\pi(n-1)/n + \delta/2 \qquad (57)$$

then $\psi_{11} + \psi_{12} = \delta$. There is a wide variety of choices for the a_j. Let n be even, and set

$$a_{n-j} = 0 \qquad (58)$$

except for $a_{n/2-1}$ and a_{n-1}. Then (57) is satisfied when

$$a_{n/2-1} = -\sin(\pi/n) + \Im(\delta/2) \;, \quad a_{n-1} = \cos(\pi/n) + \Re(\delta/2) \;. \qquad (59)$$

So if δ is small, and the a_j are selected according to (55), (58), (59) and (56), all of the first column of V will be large, with magnitude $O(1/\delta)$.

Numerical examples. Order-8 Toeplitz matrices were generated according to (55), (58), (59) and (56), with $\delta = 10^{-k}$, $k = 2, \ldots, 16$. For each matrix, the system $T\mathbf{x} = \mathbf{1}$ was solved. It was found that the normalized solution error $\|\tilde{\mathbf{x}} - \mathbf{x}\|/\|\mathbf{x}\|$ grew as the square of $1/\delta$, and the normalized residual $\|T\tilde{\mathbf{x}} - \mathbf{1}\|/\|\mathbf{b}\|$ grew linearly with $1/\delta$. Thus the algorithm is only weakly stable in this case.

6 Modified GKO Algorithm

The problem with the original pivoting strategy is that when all elements of $\mathbf{r}_{:1}$ are small and all elements of $\mathbf{v}_{:1}$ are large, normal partial pivoting will not stabilize the algorithm. Complete pivoting will do so, but requires $O(n^2)$ operations to find the pivot at each major step and $O(n^3)$ operations overall. However, a strategy of using the largest element in the first row *and column* should stabilize the algorithm in most cases, and we see that it does in the above cases.

To incorporate this row-1/column-1 pivoting, it is easy to see that the following steps should be added to the GKO algorithm (Algorithm 2.2):

- Step 1: add substep $P' \leftarrow I$, where P' will be the matrix of column interchanges.
- After loop to recover column 1 of R_k : add loop to recover row 1 of R_k.
- After loop to find maximum $\max_1$ in column 1 : add loop to find maximum $\max_2$ in row 1. If $\max_1 \geq \max_2$, carry out row interchanges as in Algorithm 2.2. Otherwise carry out column interchanges by swapping the appropriate elements in $\mathbf{s}$, $\Psi^{(k)}$ and $\mathbf{r}_{k:}^{(k)}$, and the appropriate columns in U and P'.
- After computation of L, U, P and P', the factors of R are $P^T LUP'^T$.

Results. When the modified algorithm was used on the same set of systems as was considered in the previous section, it was found that the normalized solution error $\|\tilde{\mathbf{x}} - \mathbf{x}\|/\|\mathbf{x}\|$ grew linearly with $1/\delta$ and the condition number of T, and the normalized residual $\|T\tilde{\mathbf{x}} - \mathbf{1}\|/\|\mathbf{b}\|$ was approximately constant at about 4×10^{-15}, a small multiple of ϵ. Thus the modified algorithm is stable in this case.

7 Conclusions

It has been shown that bound for the backward error in the GKO algorithm is similar to that for partial pivoting, except that extra factors, the generator growth factors, are included. These factors can be large when there is sufficient cancellation in the computation of the generators. Examples of this have been presented, and it was demonstrated that the original GKO algorithm was only weakly stable in these cases. A modified version which uses row 1/column 1 pivoting was then presented; this version was stable in these cases.

It is not known whether there are any cases upon which the modified algorithm will give large errors. Further work needs to be done to ascertain this, and if such cases can be found, the pivot strategy needs to be improved further. The aim is to find the maximum in R, or an element close to the maximum, still in $O(n)$ operations. An extension of the above strategy may be to have a few iterations in the search, i.e. search for the row-1/column-1 maximum, say at r_{1p}, then search along column p for the maximum there, and so on. This may find a better pivot at the expense of some extra work.

A practical strategy is to use the modified algorithm of §6 followed by a check of the residual; in the unlikely event that the residual is large we can resort to a stable $O(n^3)$ algorithm.

References

[1] E. Bareiss, “Numerical solution of linear equations with Toeplitz and vector Toeplitz matrices”, *Numer. Math.* 13 (1969), 404–424.

[2] A. W. Bojanczyk, R. P. Brent and F. R. de Hoog, “Stability analysis of a general Toeplitz system solver”, *Numerical Algorithms*, to appear. Preliminary version available as TR-CS-93-15, CSL, ANU, August 1993 (revised June 1994) [available by anonymous ftp from `nimbus.anu.edu.au:/pub/Brent/rpb143tr.*`].

[3] A. W. Bojanczyk, R. P. Brent, F. R. de Hoog and D. R. Sweet, “On the stability of the Bareiss and related Toeplitz factorization algorithms”, *SIAM J. Matrix Analysis Appl.* 16 (1995), 40–57.

[4] R. Brent, “Old and new algorithms for Toeplitz systems”, *Proceedings SPIE, Volume 975, Advanced Algorithms and Architectures for Signal Processing III* (edited by Franklin T. Luk), SPIE, Bellingham, Washington, 1989, 2–9.

[5] J. R. Bunch, “Stability of methods for solving Toeplitz systems of equations”, *SIAM J. Sci. Stat. Comp.* 6 (1985), 349–364.

[6] T. F. Chan and P. C. Hansen, “A lookahead Levinson algorithm for general Toeplitz systems”, *IEEE Proc. Signal Processing* 40 (1992), 1079–1090.

[7] J. Chun and T. Kailath, “Fast triangularization and orthogonalization of Hankel and Vandermonde matrices”, *Linear Alg. Apps.* 151 (1991), 199–228.

[8] I. Gohberg, T. Kailath and V. Olshevsky, “Gaussian elimination with partial pivoting for structured matrices”, preprint, May 1994.

[9] G. H. Golub and C. Van Loan, *Matrix Computations*, 2nd ed., Johns Hopkins Press, 1989.

[10] M. H. Gutknecht and M. Hochbruck, “Look-ahead Levinson and Schur algorithms for non-Hermitian Toeplitz systems”, *IPS Research Rept.* 93-11, ETH-Zürich, August 1993.

[11] D. R. Sweet, “The use of pivoting to improve the numerical performance of Toeplitz matrix algorithms”, *SIAM J. Matrix Anal. Appl.* 14 (1993), 468–493.